

Extraction of Layout Entities and Sub-layout Query-based Retrieval of Document Images

Anukriti Bansal · Sumantra Dutta Roy · Gaurav Harit

Received: date / Accepted: date

Abstract Layouts and sub-layouts constitute an important clue while searching a document on the basis of its structure, or when textual content is unknown/irrelevant. A sub-layout specifies the arrangement of document entities within a smaller portion of the document. We propose an efficient graph-based matching algorithm, integrated with hash-based indexing, to prune a possibly large search space. A user can specify a combination of sub-layouts of interest using sketch-based queries. The system supports partial matching for unspecified layout entities. We handle cases of segmentation pre-processing errors (for text/non-text blocks) with a symmetry maximization-based strategy, and accounting for multiple domain-specific plausible segmentation hypotheses. We show promising results of our system on a database of unstructured entities, containing 4776 newspaper images.

Keywords Document Image Retrieval · Sub Layout-based Matching · More

1 Introduction

The advancement of digitization of document images has increased the need for an accurate, efficient and

user-friendly large-scale information search and retrieval from the databases and archives. Document retrieval can be broadly classified into two types: content-based and layout-based ([8]). Content-based approaches are highly dependent on document labeling, feature selection and high computation involved for OCR systems. Another issue with content-based systems is that the layout information of the document is lost completely which constitute an important clue while searching for particular type of information. Besides this, a more fundamental problem with the content-based system is that the textual content of the document to be searched for is irrelevant/not known exactly. Further, the required information could be present in a small region ([2] and [13]). An example includes a portion of a newspaper image with fixed format and position, e.g., a columnists' article in the leftmost image of the editorial page in Fig. 11, highlighted in orange. Thus, the sub-layout of blocks/image-entities plays a significant role in querying for such documents. A sub-layout can be defined as an arrangement of blocks within a portion of document/image. A sub-layout-based method can enhance existing content-based retrieval by reducing the set of candidate documents [21]. (A user may seek an article about Bill Gates in the lower half of a newspaper page, which has a text block followed by an image on the left and text on the right).

A sub-layout-based retrieval can be used while designing the complete page layout of magazines, newspapers and official documents. A user may want to survey a set of designs which need some fixed content, and some part of it left to the a designer's imagination and decision. For example, the cover page of a thesis has few components fixed, such as the University logo sandwiched between text. A user can give this sub-layout as a query, and based on the retrieved results, draw fur-

A. Bansal
Indian Institute of Technology Delhi
New Delhi, India
E-mail: anukriti1107@gmail.com

S. Dutta Roy
Indian Institute of Technology Delhi
New Delhi, India
E-mail: sumantra@ee.iitd.ac.in

G. Harit
Indian Institute of Technology Jodhpur
Rajasthan, India
E-mail: gharit@iitj.ac.in

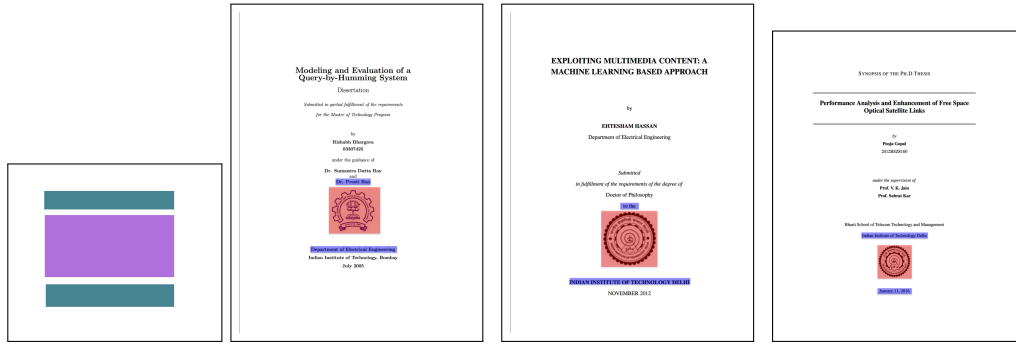


Fig. 1: Application of sub-layout-based retrieval: (a) represents the query for image region (shown in pink color) sandwiched between text regions (shown in blue color) on both the sides and present at the center of the page. (b), (c), (d) are the examples of retrieved document images.

ther cues for placing other components, to design the entire cover page (Figure 1).

Studies have shown that humans have better memory of images over text ([3], [20]). Thus, layout information can be used in applications such as personal filing system, where images are viewed and archived by an individual, and later retrieved by visual recollection. Neuroscience studies ([12], [16]) suggest that sketching is a fundamental way for humans to conceptualize and render things. This paper uses sketch-based queries so that a user can specify the desired layout/sub-layout in a natural and expressive manner.

Delivery of desired information in proper format and right quantity motivates us for sub-layout-based document image retrieval using user specified sketch-based queries.

1.1 Related Work

Image retrieval and classification has been an interesting research problem from last two decades ([21], [15], [17], [8], [6], [1]). Layout information can be used for document image classification as well as retrieval.

Hu et al. [10] compute the distance between image rows after a segmentation into a grid of equal-sized cells. Each cell is labeled as text or whitespace on the basis of its overlapping with the text block. Document images are then compared using dynamic programming on a row-based representation of documents.

Hu et al. [9] describe methods for document image classification at the spatial layout level. The elements of spatial layout are captured by a feature set called interval coding, which encodes structural layout information of the region using fixed-length vectors. These features are used in a hidden Markov model-based page layout classification system.

Shin et al. [22] use visually similar features of layout structure such as the percentage of text and non-text regions, column structures, relative font sizes, density of the content area and statistics of features of connected components, for the classification of document pages using decision tree classifiers and self-organizing maps.

Kumar et al. [11] use layout and spatial organization of document image content for image classification and retrieval. They recursively partition the image and compute histograms of codewords (SURF descriptors) in each partition. A random forest classifier is used for classification and retrieval.

The state-of-the-art in camera-based document image retrieval perhaps comes from Osaka Prefecture University Group, where Takeda et al. [23] and Nakai et al. [18] have used Locally Likely Arrangement Hashing (LLAH) for real time document retrieval. The method uses centroid of word regions as feature points and calculates geometrically invariant features on the basis of neighboring feature points. The method works well on camera captured images as queries, which contains many word images. This may not work well for queries where word blocks are not large in number, or those with a large number of non-text blocks (images, graphics) [23]. Such a method relies on the actual content, hence limiting its scope. A layout-based method may work better in such cases, which considers the relative arrangement of all types of blocks (text/non-text). Further, if we extend the LLAH idea from word blocks to centroids of text/non-text blocks, the structurally invariant information may not be adequately captured with a few layout components.

van Beusekom et al. [4] use layout information for document retrieval. A class of distance measure based on two-step procedure is introduced. In the first step, the distances between the blocks of document and query layouts are calculated. Various types of distance mea-

asures like, Manhattan distance of corner points, overlapping area of blocks, difference in width and height, etc., were used to compute distance between every pair of blocks in given layouts. Then, the matching step matches the blocks of query layout to the blocks of reference layout by minimizing the total distance. Since the method uses distance measure, it is not invariant to position and shape of the blocks. Besides, the number of blocks in case of sub-layouts will be lesser and their position and aspect ratios can be different, the total distance between two layouts will be larger and thus, may not match.

The above systems work on the basis of the layout of the complete document image, and not specific sub-layouts. Shin and Doermann [21] describe a system for sub-layout based document matching. They measure the query-to-document similarity by comparing the edges of blocks at approximately the same location in the query and the candidate image, after *uniform* scale normalization. Thus, a solution is required for the problem of sub-layout-based search where the query layout can be present at different scales and translations. Fig. 2 shows an example of query image and the retrieved image from the database using our method, whose layout entities are of different size, aspect ratio and the layout itself is at a position different from the layout in the query image. For images in this paper,

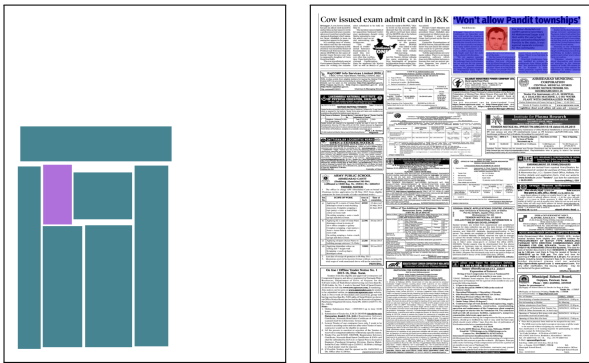


Fig. 2: Our system is invariant to scale and translation: an example. The first image is the query image and the second one is the retrieved image from the database, whose blocks are of different sizes, aspect ratios and are present at absolute positions different from those in the query

cyan/blue represents text, pink/red represent non-text non-background blocks, and grey indicates that the specific block type (text/non-text) is irrelevant.

Luqman et al. [14] present a graph-based method for document retrieval, which can perform sub-graph

search and spotting. The authors start from constructed graphs, and do not consider segmentation of images and obtaining the subsequent graph structure. The paper mentions that the overall accuracy depends on the formation of the graph structure.

1.2 Overview and Contributions

In this paper, we present a complete system for layout/sub-layout based document image retrieval based on the specification of one or more sub-layout (as a sketch-based query). Each database image is processed in an offline process to extract significant homogeneous regions of layout entities. The processed image is represented as 2D-graph, where each layout entity corresponds to graph node and edges are represented by the relationship between different nodes. The process starts with the extraction of layout features in two stages: (a) Symmetry-maximization-based pre-processing for the cases of over-segmentation, (b) Generating multiple plausible segmentation hypotheses from domain specific information. The matching algorithm uses neighborhood information as a feature in hash-based indexing for faster retrieval.

The major contributions of this paper are as follows:

1. The system is script-independent and works well even for highly unstructured documents such as newspaper pages.
2. The method works for both complete and partial document matching, non-uniform scaling and translations.
3. Modeling sketch-based sub-layout queries, including queries with missing blocks. The system handles combinations of multiple sub-layouts, specified as a Boolean query with optional approximate positional information.
4. A symmetry maximization-based scheme for cases of over-segmentation, and using multiple domain-specific segmentation hypotheses, to increase recall statistics.
5. A hashing-based strategy to prune a possibly large search space, using neighbourhood information.

The layout of the rest of the paper is as follows. Section 2 describes the preprocessing steps for the extraction of layout entities and representing a document image in the form of a graph of layout entities. This section proposes two important methods for handling the cases of over-segmentation errors and varied range of user-specified sketch-based queries. Section 3 presents various types of queries handled by our system and explains formulation with examples. Section 4 explains in detail our proposed search and retrieval procedure. This

section shows the use of context information as a feature for hash-based indexing for pruning the plausibly large search space. Section 5 shows results of successful retrieval on various types of queries, with multiple sub-layouts and on documents with script different from Latin. We conclude the paper in Section 6.

2 Pre-processing and Document Representation

The inputs to the system are scanned document images. The images are converted into gray scale and are binarized using Otsu's method [19]. Horizontal and vertical lines are identified using connected component analysis and are removed by replacing them with background pixels. Next, segment the binarized image into text and non-text regions using multi-level morphological image processing operations ([5]) (Fig. 3b). Homogeneous re-

gions of document blocks are extracted from each image. Since our dataset contains newspaper images with text of different font sizes, a simple run-length smoothing algorithm will not give homogeneous regions. To obtain proper blocks of text regions, we use Adaptive Run Length Smoothing Algorithm (hereafter, ARLSA) with a structuring element of size in accordance with the height of the connected components. The obtained text blocks are shown in Fig. 3c. Out of 5128 newspaper images of our dataset, this step gives us correct blocks in 93.136% images. A failure case involves overlapping text blocks, highlighted in red and blue in Fig. 4. Creation

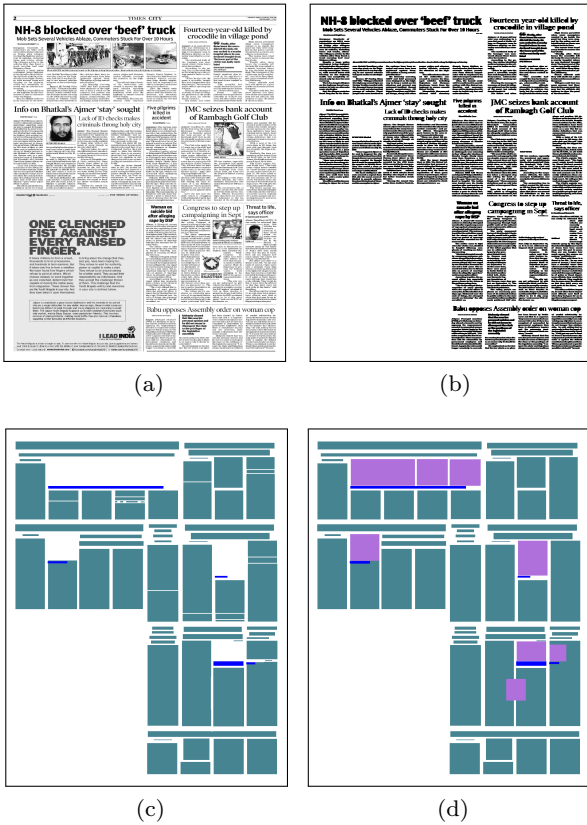


Fig. 3: (a) The original image, (b) the output of text-non-text segmentation, (c) text blocks using ARLSA (Sec. 2), and (d) the result of symmetry maximization, which groups perceptually similar and physically close blocks. (Sec. 2.1)

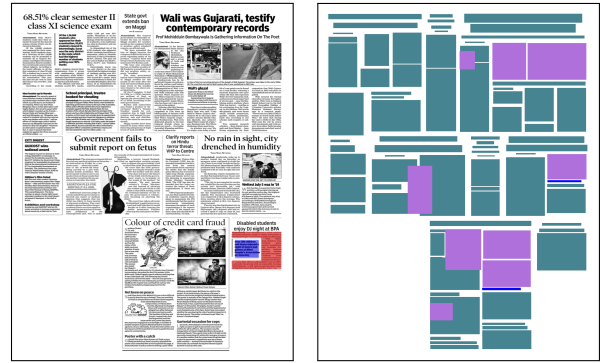


Fig. 4: ARLSA failure case (Sec. 2): In case of overlapping text blocks (shown in red color in the first image), when the minimum bounding rectangular boxes are formed (shown in second image), the blocks get merged into one.

of minimum bounding rectangles merges these blocks.

The image obtained after ARLSA can be represented as a 2-D graph. Major limitation of a graph-based representation is that they are sensitive to noise and over-segmentation errors. The paper proposes two pre-processing methods suitable for handling such issues, and going forward with multiple hypotheses corresponding to different plausible block graphs. The following sections explain these in detail.

2.1 Symmetry Maximization

The application of ARLSA gives us the block structure as in Fig. 3c. For layout and sub-layout-based retrieval, we group paragraphs into a single block by applying symmetry maximization ([25]), inspired by the Gestalt Law of Pragnanz ([24]) (which emphasizes the existence of symmetry and regularity during perceptual grouping). A symmetry-driven search is performed in the vertical direction for an optimal grouping of over-segmented blocks. Block features used for this purpose

are average character height, alignment (left, right and center), distance between two blocks, and the presence or absence of horizontal line between the blocks. We merge neighboring blocks, aligned along the top/bottom edges, if the following conditions are satisfied: (1) they are left, right or centrally aligned, (2) their average character heights are same, (3) distance between them is less than average character height of the image, and (4) no horizontal line is present between them. Fig. 3d shows the blocks obtained after this step.

2.2 Generating Multiple Segmentation Hypotheses

The graph structure obtained after the application of symmetry maximization contains blocks of different sizes, some of which are very small (e.g., author block in newspaper images). While drawing blocks for a sketch-based query, a user may not want to specify small details of the layout. *To handle such issues, the system uses domain-specific information to get a set of segmentation hypotheses.* The database stores multiple graphs corresponding to these different plausible segmentation hypotheses. Fig. 5 illustrates this technique. The first segmentation hypothesis corresponds to the output of symmetry maximization (Fig. 5a), explained in the previous section. Another possible hypothesis considers the removal of small blocks (which could be noise, or insignificant ones, such as author blocks). We identify such blocks if their height is less than or equal to the average character height of the document image, and which are sandwiched between two text blocks. Fig. 5b shows such an example. Another segmentation hypothesis has merged close-by aligned non-text blocks, as in Fig. 5c (this example has three such merged blocks). Another possible segmentation hypothesis considers removal of single line caption blocks (on the basis of their height and position with respect to neighboring blocks), as in Fig. 5d. In the database, each block is stored with its attributes and context information. For each block, we store its features such as the height, width, document ID, block ID, average character height of document, average character height of block, and spatial location in the document (top, bottom, left, right, or center). Context information consists of the block IDs of the neighboring blocks towards its four sides i.e., top, bottom, left and right. This context information is the key feature for hash-based indexing.

3 Query Formulation

We have conducted a study on 54 people to know their preferences (what and how) for layout-based searches.

On the basis of their feedback, we have created the ground-truth, designed queries and adopted a sketch-based method to specify the desired layout. Fig. 6 shows the types of queries possible in our system. Cyan represents text, and pink, image/non-text/non-background regions. Gray indicates cases when the specific block type (text/non-text) is not specified by the user. Our system performs a block arrangement-based match and retrieval. The user has an option to specify only the desired blocks in a particular sub-layout. The remaining space can be left vacant as in Fig. 6e, to indicate that a *partial match* is intended. The retrieval system can then use the vacant space to match with one/more blocks corresponding to a database document. (Fig. 8e: Sec. 5 shows an example of retrieval corresponding to the query layout in Fig. 6e). Queries can be grouped into the following categories:

1. Type 1: The type (text/non-text) is specified for all blocks, without any missing layout entities (Fig. 6a, 6b).
2. Type 2: The type is not specified for any block, and there is no missing layout entity (Fig. 6c).
3. Type 3: The type is specified for a few blocks, and there is no missing layout entity (Fig. 6d).
4. Type 4: The type is specified for all blocks, but there are some missing layout entities (Fig. 6e).
5. Type 5: No block has its type specified, and a few layout entities are missing (Fig. 6f).
6. Type 6: The type of a few blocks is specified, and some layout entities are missing (Fig. 6g).

In general, a query layout can be matched anywhere in a document. *The system also supports retrieval on the basis of a combination of multiple sub-layouts specified, along with their approximate geometric locations.* We allow such a combination to be conveniently specified using the Boolean operations *AND*, *OR* and *NOT*. Consider three sub-layouts *A*, *B* and *C*, of types 1, 2 and 6, respectively. Suppose the user has specified sub-layout *A* as in Fig. 6c, and Figs. 6a and 6g as sub-layouts *B* and *C*, respectively. The user specifies the required query as: $(A, \text{bottom}) \text{ AND } (B) \text{ AND } (\text{NOT } C)$. In other words, the user wants to search for documents which specifically have sub-layouts *A* and *B* (with *A* specifically at the bottom), but not containing sub-layout *C*. (Sec. 5, Fig. 11 shows an example of a retrieved combination of sub-layouts.) In the next section (Sec. 4), we describe the basic search and retrieval strategy for any particular sub-layout.

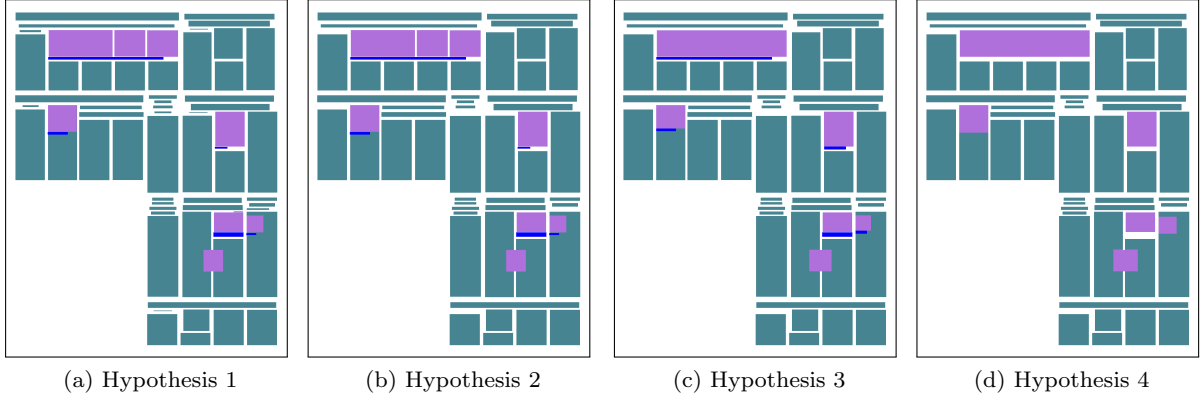


Fig. 5: Multiple segmentation hypotheses: (a) The ARLSA output (Sec. 2.1), (b) Hypothesis with small (e.g., noise) or insignificant blocks (author blocks, for instance) removed, (c) Hypothesis with grouped adjacent non-text regions, and (d) Hypothesis with caption blocks removed.

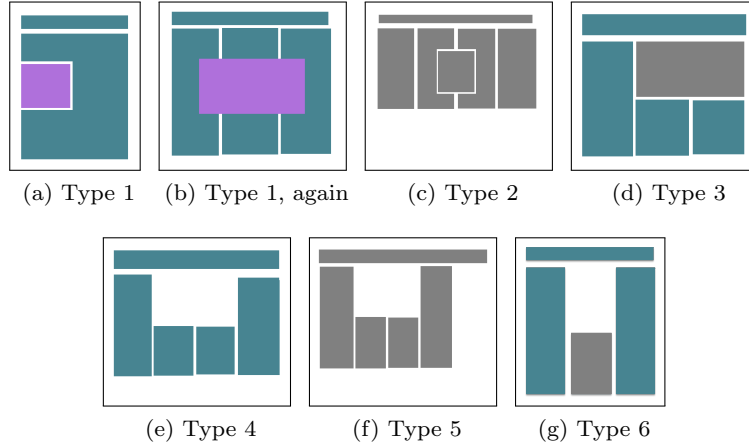


Fig. 6: Various types of query layouts: Blue represents text, pink represents non-text non-background blocks, and grey indicates that specific block type (text/non-text) is irrelevant. (a),(b) All blocks have their type specified, without any missing blocks, (c) The specific block type (text/non-text) is not relevant for any block, and there are no missing blocks. (d) Some blocks need to be retrieved without bothering about their specific type (text/non-text), and there are no missing blocks. (e) Some blocks missing, with the block type specified for all blocks. (f) Missing blocks, and block type (text/non-text) is irrelevant for all blocks. (g) Missing blocks, the type is specified for a few blocks.

4 The Proposed Search and Retrieval Procedure

The proposed graph-based matching is on the basis of the relative arrangement of blocks, irrespective of their actual dimensions. This imparts relative invariance to factors such as scale and translation. The overall steps are as below:

1. For a database document, initialize the candidate reference block set, starting with all blocks which have a similar neighborhood as the top-left block of the query (the ‘reference block’, hereafter). An important feature of our method is to prune this possibly large search space, using a Hashing-based strategy. (Sec. 4.1 has the details.)
2. Start the matching process with a node from the candidate reference block set and the query’s reference block.
 - (a) Identify if the query sub-layout corresponds to a partial match (Sec. 3). A partial match considers cases of the dimensions of a vacant space (due to missing blocks) being more than a certain fraction of the current block’s dimension (our implementation has 25%). Alternately, the dimen-

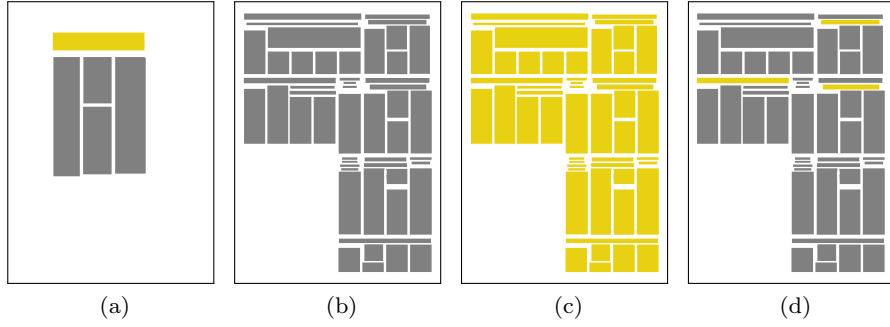


Fig. 7: (a) The reference block (highlighted in yellow) in the query image. In this example, without the Hashing-based pruning, all blocks (in (b)) get selected as candidate reference blocks (as shown in (c)). The Hashing-based pruning strategy (Sec. 4.1) helps to prune the large search space, as in (d).

sions of the vacant space should be greater than the minimum dimensions of the adjacent blocks. Fig. 6e shows a query with a partial match at the lower side of the reference block, while Fig. 6b shows a complete match.

- (b) Compare the neighborhood of the blocks in terms of relative position and block type (text/non-text). For a partial match, we insert a dummy block in the vacant space (corresponding to one or more missing blocks in the query, as in Fig. 12b) and match the neighborhood of the rest of the blocks. Fig. 12 (Sec. 5) shows final retrieval results involving such missing blocks.
- (c) Keep traversing the graph till a mismatch is found, or all blocks of the database document and/or query layout are traversed.
3. If the sub-layout of all non-dummy blocks in the query matches the sub-layout of blocks corresponding to the candidate reference block, we declare a match.
4. Go to step 2 and repeat the process with another block from the candidate reference block set, till all blocks in the set are processed.
5. The proposed system ranks the retrieved results, in order of relative average discrepancy in block aspect ratios with reference to the query sub-layouts, and their relative positions. Fig. 12 (Sec. 5) shows an example of such a ranking.

The proposed method is different from Coupled Breadth First Search (C-BFS) ([7]) in the following ways: (1) by integrating hash-based indexing, we reduce the time taken in brute force search to a large extent (explained in Sec. 4.1), (2) Unlike C-BFS, our procedure is capable of handling partial matches. These situations are practically quite common, when a user does not remember the exact layout, or there are some blocks missing in the

query. (3) Our system also ranks the retrieved results, as explained above.

4.1 Hashing-based Pruning of the Search Space

A brute force-based search of all candidate blocks (as in C-BFS ([7]) for instance), needs two starting nodes (one from query and one from database) as reference blocks. Since the size of the database is very large, performing one-to-one matching of blocks is not feasible. In order to narrow down the portion of database to be explored during the search, we use a Hashing-based technique. The indexing function predicts the subset of the database that needs to be searched for each query image. The context string is extracted for each block, which is used as the hash key k . A context string is a fixed-length descriptor that captures the information of a block (block-type and spatial location in the document: top, bottom, left or right), and its neighborhood. The neighborhood information of a block includes the number of blocks in all four directions (top, bottom, left and right) and whether they are overlapping with the block. A hash value is calculated from it using the equation $H = k \bmod n$, where n is the number of bins in the hash table. The block ID and database document ID is stored in the corresponding hash bin. Chaining is applied to resolve collisions that occur when two keys map to the same hash bin.

Fig. 7 illustrates how hashing prunes the search space. Fig. 7a shows a query layout in which reference node is shown in yellow color. Fig. 7b is a sample document image containing 53 blocks. A brute force technique without hashing will need to consider each of the 53 blocks as a candidate reference block, as shown in Fig. 7c. A good hash-function will put all the blocks with similar block and context information in the same bucket. During searching and retrieval, when we match the hash-



Fig. 8: Retrieved Results: (a) Result for query layout shown in Fig. 6b, (b) Result for query layout shown in Fig. 6c. For the query layout shown in Fig. 6d, the gray block substituted by non-text block (pink color) in (c) and by text block (blue color) in (d). (e) shows the partial matching result for query layout shown in Fig. 6g. (f) shows retrieved results for query layout shown in Fig. 6f: the middle missing block substituted by a non-text one, and the ones to the right, as three text blocks.

value generated using the context string of the query block’s reference node, the number of elements in candidate reference block set gets reduced to 3 (shown in Fig. 7d in yellow).

5 Results and Discussion

In the absence of a suitable publicly available large dataset, we created a dataset of newspaper images: a challenging domain, since they differ widely in their page layouts, and subsume more ordered layouts such as in journals and books, and non-rectangular mixed layouts in some magazines. ARLSA generates correct blocks on 4776 of 5128 images. (Details in Sec. 2, including a failure case example.) We create a graph database for these 4776 images, using symmetry maximization (Sec. 2.1) and multiple segmentation hypotheses (Sec. 2.2).

5.1 Handling different types of Queries

Fig. 8 summarizes experimental results with various types of query layouts (Fig. 6). (We describe a case of retrieval with the sub-layout in Fig. 6a specifically in the context of simultaneous retrieval of multiple sub-layouts, later in this section). A retrieval example for the sub-layout Type 1b (Fig. 6b): block types completely specified, no blocks) is in Fig. 8a. For sub-layout Type 2 (Fig. 6c: no block type specified, no missing blocks), Fig. 8b shows an example of successful retrieval. In the query layout of Type 3 (Fig. 6d), the gray block indicates that the block type (text/non-text) has not been specified. The retrieval result in Fig. 8c shows the gray block substituted by a non-text block (highlighted in pink), In Fig 8d, it is substituted by a text block (highlighted in blue). Query Type 4 (Fig. 6e) has all block types specified but missing blocks. Fig. 8e shows a retrieval result with the same sub-layout found in two places in the same document: the top one has one

missing non-text block, while the bottom one has four missing blocks (3 text and one non-text). All missing blocks in Fig. 8e are shown in gray. In Query Type 5 (Fig. 6f), the type (text/non-text) is not specified for any block, and there are one or more missing blocks, which may be spatially located anywhere. Fig. 8f shows a retrieved result, which has the middle missing block as a non-text one, and the ones to the right, as three text blocks. Query Type 6 (Fig. 6g) considers cases with the block type specified for a few blocks, and missing blocks. Fig. 9 shows an example of successful retrieval interestingly, for a newspaper in a different language and script (Hindi, Devanagari). *The system is independent of the specific language and the script, since the retrieval is on the basis of the relative arrangement of blocks.* Fig. 10 shows results of successful retrieval of

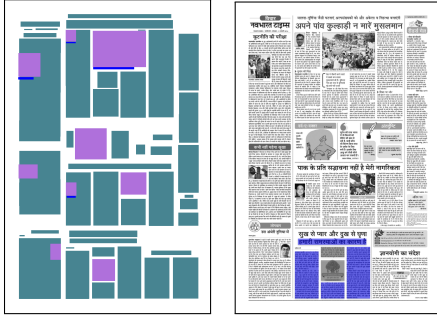


Fig. 9: The system is language and script-independent since it considers the relative arrangement of blocks. For a Hindi newspaper in Devanagari script, the images above show the block structure (Sec. 2), and results of successful retrieval (Query Type 6, Fig. 6g).

layouts with lines of text interspersed with parts of images. (These correspond to Fig. 8a and Fig. 8b. The system considers the minimum rectangular bounding boxes for the irregular images. As mentioned in Sec. 2 (Fig. 4), the system fails as the pre-processing module is not able to handle overlapping text blocks. As shown in Fig. 10, the system can handle overlapping blocks of different types (text and non-text).

5.2 Handling Combinations of Multiple Sub-layouts

As mentioned in Sec. 3, the system supports retrieval of a combination of multiple sub-layouts. Fig. 11 shows the retrieval results for the query (*A, bottom*) AND (*B*) AND (*NOT C*). The sub-layouts A, B, and C correspond to query layouts in Fig. 6b, 6a, and 6g, respectively. The left-most document in Fig. 11 is the correctly retrieved document which satisfies all 4 constraints. The middle one



Fig. 10: Handling irregular layouts: the two examples above correspond to lines of text interspersed with images of irregular dimensions. The system considers the closest rectangular bounding boxes and is able to perform correct retrieval: these correspond to Figs. 8a and 8b.

Table 1: Statistics of search and retrieval procedure

Query	# Documents	Recall (%)	Precision (%)	Time(sec)
Type 1	1592	95.41	98.14	0.685
Type 2	1986	95.21	97.91	0.696
Type 3	1986	95.21	97.91	0.696
Type 4	2990	91.42	98.387	0.742
Type 5	2990	91.42	98.387	0.742
Type 6	2990	91.42	98.387	0.742

is not retrieved since it violates the spatial constraint for sub-layout A. The right-most is not retrieved since the presence of sub-layout C violates the query specifications.

5.3 Ranking of Results, and Retrieval Statistics

Fig. 12 shows an example of ranked results for a Type 4 query layout (Fig. 12a). As mentioned earlier (Sec. 4), we create a dummy block (Fig. 12b) in the vacant space, which corresponds to one or more missing blocks. The matched layouts are ranked on the basis of relative average discrepancy in block aspect ratios, and relative positions, compared to the query blocks. Table 1 lists recall and precision statistics on the database of 4776 images, for various types of query layouts. Recall and precision values are calculated on the basis of the number of documents retrieved ('# Documents'), and not (NOT) total number of layouts. (A document may contain more than one instance of a particular type of query sub layout, as in Fig. 12c.) The departure from perfect statistics is on account of pre-processing errors (in Symmetry maximization (Sec. 2.1) and generating multiple

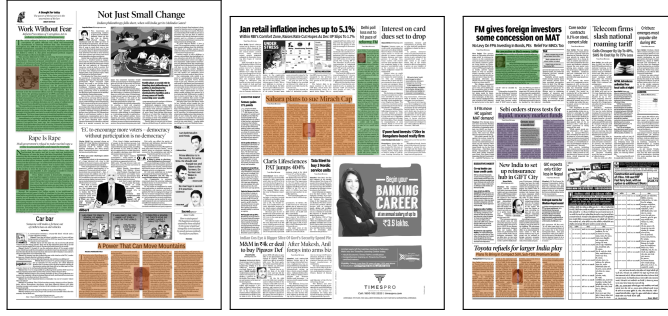


Fig. 11: Retrieval results for a combination of multiple sub-layouts in different spatial locations (Sec. 3). Query (A, bottom) AND (B) AND $(NOT C)$ (Sec. 5.2) results in left-most result alone, and not the other two. The middle one does not satisfy the spatial location requirements for sub-layout A , whereas the rightmost one also contains layout C .

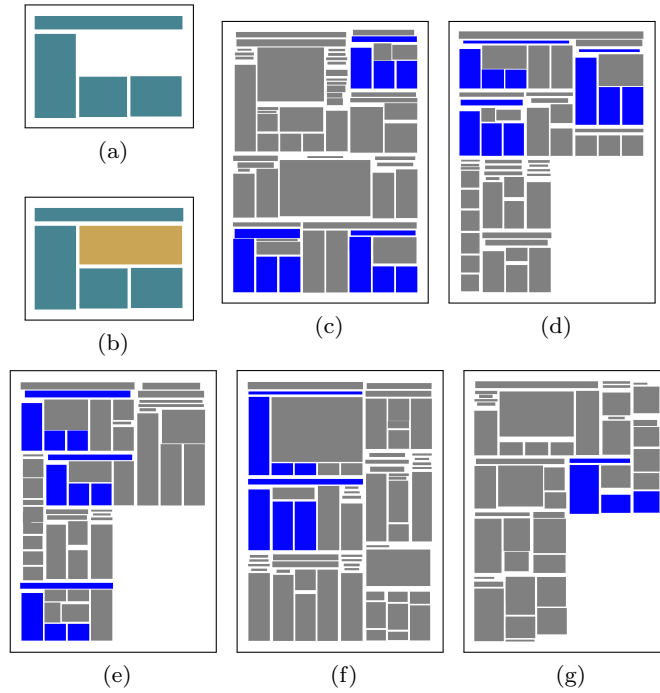


Fig. 12: Ranked results for the query layout in (a): The system first generates a dummy block (Sec. 4). The system ranks results in order of relative average discrepancy in block aspect ratios, and relative positions (Sec. 4.)

segmentation hypotheses (Sec. 2.2). To establish the utility of generating multiple segmentation hypotheses, an experiment omitted this stage completely. The precision value in this case was same, i.e., 98.1%, but recall fell to less than 40%. Reasons include missing out on plausible hypotheses, and the presence of small and/or noisy blocks which unnecessarily disturb the document graph structure (Sec. 2).

6 Conclusions

We propose a system to retrieve user-specified combinations of sub-layouts, which works for highly unstructured documents such as newspaper images. The graph-based matching strategy integrates a hashing-based indexing for fast matching, with handling cases of pre-processing segmentation errors. We show encouraging retrieval results for a representative sample database of 4776 newspaper images.

References

1. Antani S, Kasturi R, Jain R (2002) A survey on the use of pattern recognition methods for abstraction, indexing and retrieval of images and video. *Pattern Recognition* 35(4):945–965
2. Aurisicchio M, Langdon PM, Wallace KM (2003) Investigating knowledge searches in aerospace design. In: *ICED03, 14th International Conference on Engineering Design*, Stockholm, Sweden, pp 293–294
3. Baggett P (1979) Structurally equivalent stories in movie and text and the effect of the medium on recall. *Journal of Verbal Learning and Verbal Behavior* 18(3):333 – 356
4. van Beusekom J, Keyzers D, Shafait F, Breuel T (2006) Distance measures for layout-based document image retrieval. In: *Proc. Int'l Conf. on Document Image Analysis for Libraries (DIAL)*, pp 11 pp.–242
5. Bloomberg DS (1991) Multiresolution morphological approach to document image analysis. In: *Proc. Int'l Conf. on Document Analysis and Recognition (ICDAR)*
6. Chen N, Blostein D (2007) A survey of document image classification: Problem statement, classifier architecture and performance evaluation. *International Journal on Document Analysis and Recognition* 10(1):1–16
7. Chikkerur S, Cartwright AN, Govindaraju V (2006) K-plet and coupled bfs: A graph based fingerprint representation and matching algorithm. In: *Proceedings of the 2006 International Conference on Advances in Biometrics, ICB'06*, pp 309–315
8. Doermann DS (1998) The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding* 70(3):287–298
9. Hu J, Kashi R, Wilfong G (1999) Document classification using layout analysis. In: *Database and Expert Systems Applications, 1999. Proceedings. Tenth International Workshop on*, pp 556–560
10. Hu J, Kashi R, Wilfong G (1999) Document image layout comparison and classification. In: *Proc. Int'l Conf. on Document Analysis and Recognition (ICDAR)*, pp 285–288
11. Kumar J, Ye P, Doermann DS (2014) Structural similarity for document image classification and retrieval. *Pattern Recognition Letters* 43:119–126
12. Landay J, Myers B (2001) Sketching interfaces: toward more human interface design. *Computer* 34(3):56–64
13. Lowe A, McMahon C, Shah T, Culley S (1999) A method for the study of information use profiles for design engineers.
14. Luqman MM, Ramel JY, Llads J, Brouard T (2013) Fuzzy multilevel graph embedding. *Pattern Recognition* 46(2):551–565
15. Marinai S, Miotti B, Soda G (2011) Digital libraries and document image retrieval techniques: A survey. In: *Learning Structure and Schemas from Documents*, vol 375, pp 181–204
16. Marr D (1982) *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc.
17. Mitra M, Chaudhuri BB (2000) Information retrieval from documents: A survey. *Information Retrieval* 2(2-3):141–163
18. Nakai T, Kise K, Iwamura M (2006) Use of affine invariants in locally likely arrangement hashing for camera-based document image retrieval. In: *Proc. Int'l Workshop. on Document Analysis Systems (DAS)*, pp 541–552
19. Otsu N (1979) A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on Systems, Man and Cybernetics* 9(1):62–66
20. Paivio A, Csapo K (1973) Picture superiority in free recall: Imagery or dual coding? *Cognitive Psychology* 5(2):176 – 206
21. Shin C, Doermann DS (2006) Document image retrieval based on layout structural similarity. In: *Proc. Int'l Conf. on Image Processing, Computer Vision and Pattern Recognition (IPCV)*, pp 606–612
22. Shin C, Doermann DS, Rosenfeld A (2001) Classification of document pages using structure-based features. *International Journal on Document Analysis and Recognition* 3(4):232–247
23. Takeda K, Kise K, Iwamura M (2011) Real-time document image retrieval for a 10 million pages database with a memory efficient and stability improved LLAH. In: *Proc. Int'l Conf. on Document Analysis and Recognition (ICDAR)*, pp 1054–1058
24. Wertheimer M (1923) Untersuchungen zur lehre von der gestalt. ii. *Psychologische Forschung* 4(1):301–350
25. Zhang H, Xu K, Jiang W, Lin J, Cohen-Or D, Chen B (2013) Layered analysis of irregular facades via symmetry maximization. *ACM Transactions on Graphics* 32(4):121:1–121:13